

Attorney's Docket No. 009683-353

THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of)

Mitsuru Obara et al.)

Application No.: 09/427,114)

Filed: October 26, 1999)

For: ASYNCHRONOUS IMAGE DATA)
PROCESSING DATA)

Group Art Unit: 2183

Examiner: TONIA L. MEONSKE

Appeal No.:

APPEAL BRIEF

Mail Stop APPEAL BRIEF – PATENTS

Date: January 27, 2005

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

This appeal is from the decision of the Primary Examiner, dated September 2, 2004 (Paper No. (102504), in which claims Claims 1-26 were finally rejected. Claims 1-12 are reproduced as the Claims Appendix of this brief. Additionally, a copy of Figures 1-19 are attached in a drawing Appendix.

☒ A check covering the ☐ \$250.00 (2402) ☒ \$500.00 (1402)

Government fee is filed herewith.

☐ Charge ☐ \$250.00 (2402) ☐ \$500.00 (1402) to Credit Card. Form PTO-2038 is attached.

The Commissioner is hereby authorized to charge any appropriate fees under 37 C.F.R. §§1.16, 1.17, and 1.21 that may be required by this paper, and to credit any overpayment, to Deposit Account No. 02-4800.



Table of Contents

	Page
I. Real Party in Interest.....	2
II. Related Appeals and Interferences.....	2
III. Status of Claims.....	2
IV. Status of Amendments.....	2
V. Summary Claimed Subject Matter.....	2
VI. Grounds of Rejection to be Reviewed on Appeal.....	9
VII. Applicants arguments against the rejection of the claims.....	9
A. Rejection of Claims 1, 8-11, 18-24	9
B. Rejection of Claim 25	12
C. Rejection of Claim 26.....	12
D. Rejection of Claims 2 and 12	13
E. Rejection of Claims 3 and 13.....	13
F. Rejection of Claims 4 and 14.....	14
G. Rejection of Claims 5 and 15	14
H. Rejection of Claims 6 and 16	14
I. Rejection of Claims 7 and 17.....	15
VIII. Conclusion	15
Appendix A – Claims 1-26	
Appendix B – Figures 1-19	



I. Real Party in Interest

The present application is assigned to Minolta Co., Ltd., who is the real party in interest.

II. Related Appeals and Interferences

There are no known currently pending related appeals or interferences in the subject application.

III. Status of Claims

Claims 1-6 remain pending in the subject application and are being appealed.

IV. Status of Amendments

No amendments have been made to claim 1-26 subsequent to the final rejection.

V. Summary Claimed Subject Matter

The present invention relates generally to a data processing system, and more particularly, to a data processing system executing a plurality of processings in a prescribed order using a plurality of processors. (page 1, lines 8-10)

Referring to Fig. 1, the data processing apparatus includes an image input device 8 to input image data, processing portions 9 to 12 to perform various processings for each pixel data of the input image data, an image output device 13 formed of an electrophotographic printer or inkjet printer to output the processed image data onto a recording medium such as paper, and a memory 14. (page 5, lines 24-29)

Processing portion 9 performs Log conversion processing for each pixel of the image data input by image input device 8. Processing portion 10 performs MTF correction to data after the Log conversion at processing portion 9. Processing

portion 11 performs gamma correction to the data after the MTF correction at processing portion 10. Processing portion 12 binarizes the data after the gamma correction at processing portion 11. (page 5, line 30 through page 6, line 3)

Image input device 8, processing portions 9 to 12, and image output device 13 (hereinafter referred to "processing portions 8 to 13") are connected to memory 14 through a data bus, and each of processing portions 8 to 13 writes/reads data to/from memory 14 through the data bus. Memory 14 is a common memory to/from which processing portions 8 to 13 can write/read data. Memory 14 also has a controller (not shown) so that any one of processing portions 8 to 13 can read or write data. (page 6, lines 9-15)

The state flag will be now described. The state flag represents which ones of the processing by processing portions 8 to 13 the pixel data has been through, in other words the flag represents which processing is to be performed next. Fig. 3 is a table for use in illustration of the state flag. The state flag is represented by a 3-digit binary number, in other words by 3 bits. If the state flag is "000", the flag represents that the pixel data stored in the data region is data input by image input device 8 and data which can be subjected to Log conversion by processing portion 9. If the state flag is "001", the flag represents that the pixel data stored in the data region has been subjected to Log conversion, and can be subjected to MTF correction at processing portion 10. Similarly if the state flag is "010", the data has been subjected to MTF correction and can be subjected to gamma correction. If the state flag is "011", the data has been subjected to gamma correction and can be binarized. If the state flag is "110", the data has been binarized and can have its image output. If the state flag is "111", the data has its image output. (page 6, line 28 through page 7, line 10)

The state of image data stored in memory 14 will be now described. Fig. 5A shows that image data has been input by image input device 8 and stored in memory 14. The flag regions of the pixel data pieces in this state are all stored as "000". Fig. 5B shows that part of the image data stored in memory 14 has been subjected to Log conversion processing at processing portion 9. The flag state region of pixel

data subjected to the Log conversion has been changed to "001" and stored. Fig. 5C shows that part of the image data stored in memory 14 has been subjected to MTF correction at processing portion 10. The state flag regions of pixel data pieces which have been subjected to the MTF correction are stored as "010". Fig. 5D shows that part of the image data stored in memory 14 has been subjected to gamma correction at processing portion 11. The state flag regions of the pixel data pieces which have been subjected to the gamma correction are stored as "011". Fig. 5E shows that part of the image data stored in memory 14 has been binarized at processing portion 12. The state flag regions of the pixel data pieces which have been binarized are stored as "110". Fig. 5F shows that all the pixel data pieces in the image data stored in memory 14 have been binarized at processing portion 12. The state flag regions of all the pieces of the pixel data are stored as "110". (page 7, line 18 through page 8, line 3)

Referring to Fig. 6, processing portions 9 to 12 read the pixel data pieces stored in memory 14 in the order in which they have been read at image input device 8 (step S01). The state flag of the read pixel data is checked (step S02). The checking of the state flag is performed to determine whether the read pixel data can be processed. For example, referring to Fig. 3, in Log conversion, if the state flag is "000", the data can be processed, and otherwise the data cannot be subjected to Log conversion. Similarly, MTF correction can be performed only if the state flag is "001", gamma correction can be performed only if the state flag is "010", and binarization can be performed only if the state flag is "011". (page 8, lines 7-16)

If read pixel data cannot be processed (NO in step S02), after standing by for a prescribed time period (step S03), pixel data is once again read (step S01). This is because the order of processings to one piece of pixel data is prescribed and pixel data is processed in the order the data has been read by image input device 8. As a result, if the data is determined to be unable to be processed based on the result of checking the state flag in step S02, this means that the pixel data has not been subjected to processing in the preceding stage. For example, when processing portion 11 which performs gamma correction checks the state flag by reading pixel data, the state flag representing the data unable to be processed is "000" or "001".

In this case, pixel data needs only be read after the preceding processing has been completed. The prescribed time period in step S03 needs only be the time period necessary for such preceding processing. (page 8, lines 17-30)

If the processing is determined possible by the checking of the state flag (YES in step S02), the processing is executed (step S04). After the processing, the processed data and state flag are written into memory 14 (step S05). Referring to Fig. 3, the state flag written here is "001" if the process executed in step S04 is Log conversion, "010" for MTF conversion, "011" for gamma correction and "110" for binarization. (page 8, line 31 through page 9, line 3)

Then, the presence/absence of pixel data to be processed is determined (step S06), and if the pixel data to be processed is present, the control proceeds to step S01, and the above process is repeated, and if there is no pixel data to be processed, the processing ends. (page 9, lines 4-7)

In this embodiment image data is stored in memory 14 in the format having the state flag region and data region for each pixel data piece (see Fig. 2), but one state flag may be provided for a plurality of pieces of pixel data, and a format having one state flag region and a plurality of data regions may be employed. Fig. 8 shows an example of such a format having one state flag region and a plurality of data regions. The format shown in Fig. 8 is effective for example if one state flag is provided for one line of pixel data pieces, or the image data is divided into 3×3 or 5×5 matrices and the pixel data included in each matrix is provided with one flag. (page 9, lines 24-33)

Referring to Fig. 9, a data processing apparatus according to a second embodiment of the invention includes a state control portion 20 in addition to the construction according to the first embodiment. State control portion 20 is connected to an image input device 8, processing portions 15 to 18, and an image output device 13, and controls these elements. Other than the processings by state control portion 20 and processing portions 15 to 18, the data processing apparatus according to the second embodiment is the same as the data processing apparatus

according to the first embodiment, and the description is not repeated here. (page 10, lines 15-23)

Referring to Fig. 10, state control portion 20 rewrites the state flag region of memory 14 into "000" to initialize the state flag (step S10). Then, state control portion 20 constantly monitors the state of memory 14 described in conjunction with Fig. 5, and transmit to processing portions 15 to 18 the address of pixel data to be processed (step S11). Processing portions 15 to 18 access memory 14 based on the address received from state control portion 20 to read pixel data and perform respective processings. When processing has been completed at any of processing portions 15 to 18, an end signal is transmitted to state control portion 20. State control portion 20 is in a stand-by state until the end signal from processing portions 15 to 18 is received (step S12), and once the end signal is received from any of processing portions 15 to 18, the state flag region corresponding to the pixel data processed by the processing portion which has transmitted the end signal is rewritten (step S13). (page 10, line 24 through page 11, line 4)

It is then determined if the final pixel data, in other words, the data which has been read in the end by image input device 8 has the flag "110" (step S14), and if the flag is "110", the control proceeds to step S15. The control otherwise proceeds to step S11 and the process from steps S11 to S13 is repeated. (page 11, lines 10-14)

The state flag of the final pixel data having "110" means that all the pieces of pixel data have been binarized, in other words that all the processings have completed. (page 11, lines 15-17)

In step S15, an instruction to output image data stored in memory 14 is provided to image output device 13. Once image data stored in memory 14 has been printed and output by image output device 13, the flag regions of all the pieces of pixel data stored in memory 14 are rewritten from "110" into "111" (step S16). Then, the process is completed. (page 11, lines 18-22)

Referring to Fig. 11, processing portions 15 to 18 wait for an instruction from state control portion 20 (step S20). The instruction from state control portion refers

to transmission of the address of pixel data to be transmitted from state control portion 20 in step S11 in the state control processing shown in Fig. 10. When the address from state control portion 20 is received, the address in memory 14 is accessed and image data is read (step S21), and the processing is executed (step S22). The processing herein refers to Log conversion, MTF correction, gamma correction or binarization. (page 11, lines 23-31)

Once the processing to the read image data has been completed, the processed data is written in memory 14 (step S23). The address to which the data is written at this time is the address received from state control portion 20 in step S20. Once the writing to memory 14 is completed, an end signal is transmitted to state control portion 20 (step S24). (page 11, line 32 through page 12, line 3)

Thus, the data processing apparatus according to the second embodiment controls processing portions 15 to 18 as it monitors the progress in processing portions 15 to 18 by state control portion 20, so that processing portions 15 to 18 can be operated asynchronously, i.e., without synchronization. (page 12, lines 4-8)

Referring to Fig. 12, a data processing apparatus according to a third embodiment of the invention includes a region determining portion 30 in addition to the data processing apparatus according to the second embodiment. (page 12, lines 10-13)

Region determining portion 30 determines whether or not pixel data input at image input device 8 is pixel data of a solid image before Log conversion by processing portion 15. (page 12, lines 16-18)

The region determining process performed at region determining portion 30 will be now described. Referring to Fig. 13, the region determining process is in a stand-by state until there is an instruction from state control portion 20 (step S40). The instruction from state control portion 20 herein refers to the reception of the address of pixel data output by state control portion 20 in step S11 in the state control processing in Fig. 10. Once the address of image data is received from state control portion 20 (YES in step S40), a pixel data piece corresponding to the received address and pixel data pieces around that pixel data piece, for example the

pixel data pieces included in a 3×3 matrix having in the center the pixel data piece corresponding to the received address is read from memory 14 (step S41). (page 12, lines 19-30)

It is then determined if the 3×3 matrix region is a solid image based on the pixel image data (step S42). The solid image refers to the image in which all the pixel data included in the 3×3 matrix take the same value since image data input by image input device 8 is monochrome according to this embodiment. Note that if the image data input by image input device 8 is color data, the solid image refers to image data in which the chroma and brightness of the image data included in the 3×3 matrix both take the same value. (page 12, line 31 through page 13, line 5)

If it is determined in step S42 that the data is solid image data, a rewriting signal is output to state control portion 20 (step S44). If it is determined that the data is not solid image data (NO in step S42), a rewrite-not-necessary signal is output to state control portion 20 (step S43). Then this processing is completed. (page 13, lines 6-10)

State control portion 20 performs state control processing shown in Fig. 10, and a rewrite signal or rewrite-not-necessary signal is received from region determining portion 30 rather than an end signal. If a rewrite signal is received, the state flag is rewritten into "110" in step S13. If a rewrite-not-necessary signal is received in step S12, the state flag is rewritten into "001" in step S13. (page 13, lines 11-16)

Referring to Fig. 14, in the region determining processing, if the data is determined to be solid image data, the state flag is rewritten into "100". The image data having the state flag rewritten into "100" is then subjected to binarizing. (page 13, lines 17-20)

As described above, the data processing apparatus according to the third embodiment determines if pixel data is solid image data by region determining portion 30. If the pixel data is solid image data, three processings, Log conversion, MTF correction, and gamma correction are not performed, in other words the

intermediate process can be omitted so that the data processing can be performed at a higher speed. (page 13, lines 21-26)

VI. Grounds of Rejection to be Reviewed on Appeal

The issue presented for review by the Board of Patent Appeals and interferences is whether claims 1-26 were properly rejected under 35 U.S.C. §102(b) as being anticipated by *Kuo et al.* (U.S. Patent No. 5,299,309).

VII. Applicants arguments against the rejection of the claims.

A. Rejection of Claims 1, 8-11, 18-24

Applicants respectfully submit that the prior art does not show, teach or suggest a plurality of processors for executing a series of different types of processing functions on image data that consists of a plurality of pixel data as claimed in claims 1 and 11.

Kuo et al. appears to disclose a graphics control system for a computer system which improves the processing efficiency of the computer, especially when the computer is displaying windows. (col. 1, lines 9-12) As shown in FIG. 1, this conventional graphics control system comprises a host computer 10, a graphics processor 20, a display memory 30, and a display device, such as CRT 32. The host computer 10 comprises a main memory 12 and a CPU 14. All commands relating to the windows display are generated by the host computer 10 which communicates with the graphics processor 20 via the bus 16. The graphics processor 20 is comprised of a processing unit 22, a graphics context 24, and a drawing unit 26. The processing unit 22 performs two functions: 1) it receives and executes the commands issued by the host computer 10; and 2) it converts certain graphics parameters stored in the graphics context 24 into display data. (col. 1, lines 20-33) The graphics context 24 is a buffer that stores a set of image parameters needed to carry out a current command. For example, to draw a line segment on the CRT 32, the graphics context 24 stores the initial and final addresses of the line segment on the CRT, information relating to foreground color and background color, information

relating to the mix of the three primary colors red, green, and blue, and other similar parameters. When displaying windows, other parameters will also need to be stored, such as the manner in which overlapping images will be processed (i.e., whether by AND or NOR processing), the initial and final addresses of the windows on the CRT, etc. (col. 1, lines 40-52). In processing the content of a window image, the CPU 14 will compute the relevant graphics context parameters, enter them one by one into the graphical context 24 and instruct the graphics processor 20 to execute the drawing command (col. 1, lines 56-60). The improved graphics control system comprises a shared memory which is directly accessible by both the host computer and by the graphics processor. Because the shared memory is directly accessible by the host computer and by the graphics processor, the host computer can write the graphics context parameters of a next command into the shared memory while the graphics processor is executing a current command. When the graphics processor completes execution of the current command, it can receive the graphics context parameters of the next command to be executed directly from the shared memory. Because the host computer is able to store the next set of graphics context parameters in the shared memory while a current command is being executed, the host computer does not have to wait until completion of the current command before computing the graphics context parameters for the next command and entering them in the graphics processor. Thus, the host computer will not have to access the graphics processor so frequently when the graphics context needs to be changed. (col. 2, lines 46-68)

Thus, *Kuo et al.* merely discloses a CPU 14 which generates commands relating to a windows display (column 1, lines 24-27, column 3, lines 58-60). In addition, the CPU 14 computes the relevant graphics context parameters (column 1, lines 56-60) such as the initial and final addresses of the line segment, information relating to foreground color and background color, information relating to the mix of the three primary colors, how overlapping images will be processed and initial and final addresses of the windows (column 1, lines 42-52). Nothing in *Kuo et al.* shows, teaches or suggests a plurality of processors for executing processing functions on image data that consists of a plurality of pixel data as claimed in claims 1 and 11.

Rather, *Kuo et al.* merely discloses a CPU 14 which computes relevant graphics context parameters directed to the window display including initial and final addresses of a line segment, information relating to foreground color and background color, etc.

Applicants respectfully traverse the Examiner's statement that graphics context parameters of *Kuo et al.* are pixel data. Applicants respectfully point out that pixel data is directed to the (actual) image data itself such as output by a photoelectric conversion element. The information computed by CPU 14 represents window information on how to display pixel data. Nowhere in *Kuo et al.* does it show, teach or suggest that CPU 14 actually manipulates the pixel data itself. *Kuo et al.* merely computes window information on how to display an image such as foreground and background color, etc. In other words, the (actual) image data (pixel data) itself is not processed in *Kuo et al.* Rather, *Kuo et al.* merely discloses computing graphics context parameters in order to display an image on a display device.

Furthermore, *Kuo et al.* merely discloses a processing unit 22 which receives and executes commands issued by the host computer and converts certain graphics parameters stored in the graphics context 24 into display data (column 1, lines 29-33). Thus similarly, *Kuo et al.* does not show, teach or suggest that each processor executes a processing function different from one another on pixel data. Rather, *Kuo et al.* merely discloses that the graphics parameters generated by the CPU 14 are converted by processing unit 22 and not the (actual) image data which consists of a plurality of pixel data (such as from a photoelectric conversion element).

Applicants respectfully point out that claims 1 and 11 claim a plurality of processors for executing a series of different types of processing functions on image data that consists of a plurality of pixel data. In other words, the processing functions are executed on the pixel data itself. However, *Kuo et al.* clearly is not processing the pixel data itself, as clearly shown in Figure 3 of *Kuo et al.*, which discloses how an entire block of data is moved from one location to another location based upon the graphics context parameters calculated by CPU 14. In other words,

Kuo et al. is only directed to how to move the pixel data around the display screen and is not involved in the processing of the pixel data itself.

B. Rejection of Claim 25

As discussed above, *Kuo et al.* is only directed to the calculation and manipulation of graphics context parameters in order to display an image on a display device. Nothing in *Kuo et al.* shows, teaches or suggests a first processor executing first processing on data to be processed and a second processor for executing second processing on the data that was subjected to the first processing as claimed in claim 25. *Kuo et al.* merely discloses a CPU 14 which computes relevant graphics context parameters and processing unit 22 which converts the graphics context parameters generated by CPU 14. In other words, *Kuo et al.* takes (undisclosed) information X and computes relevant graphics context parameters (information Y) and then processing unit 22 converts the graphics context parameters (information Y) into display data. However, as claimed in claim 25, the first processor executes first processing on data while the second processor executes second processing on (the same) data subjected to the first processing. Thus, nothing in *Kuo et al.* shows, teaches or suggests a first processor for executing first processing on data and a second processor for executing second processing on the (same) data that was subjected to the first processing as claimed in claim 25.

C. Rejection of Claim 26

Similar to the above arguments, *Kuo et al.* merely discloses computing relevant graphics context parameters in CPU 14 and subsequently converting the graphics parameters by a processing unit 22. The relevant graphics context parameters are not image data but in fact relate to the initial and final addresses of a line segment, foreground and background color, etc. Furthermore, the computation of these parameters is not image processing but is in fact a calculation. Applicants respectfully point out that a calculation of data is different from processing data, since processing refers to the examination and analysis of data, including storing,

updating, combining and rearranging, whereas calculation refers to mathematical determination.

Furthermore, as discussed above, *Kuo et al.* merely discloses computing relevant graphics contexts parameters and subsequently converting them into display data. Thus, nothing in *Kuo et al.* shows, teaches or suggests a first image processor for executing first image processing on image data and a second image processor for executing second image processing on the image data that was subjected to the first image processing as claimed in claim 26.

D. Rejection of Claims 2 and 12

Kuo et al. merely discloses at column 2, lines 46-65, that the host computer 10 can write the graphics context parameter of a next command into the shared memory 34 while the graphics processor 20 is executing a current command. Nothing in *Kuo et al.* shows, teaches or suggests that each processor determines if data to be processed can be processed based on state information representing the processing to be performed next as claimed in claims 2 and 12.

Nowhere in *Kuo et al.* is it shown, taught or suggested that each processor determines if data to be processed can be processed based upon the state information representing the processing to be performed next. *Kuo et al.* merely discloses that a host computer 10 can write parameters of a next command into a shared memory 34 while the graphics processor 20 is executing a current command.

E. Rejection of Claims 3 and 13

Kuo et al. merely discloses at column 2, lines 46-65 and column 4, lines 20-24, that the host computer 10 can compute and store graphics contexts parameters for the next drawing command while a current drawing command is being executed. Nothing in *Kuo et al.* shows, teaches or suggests that each of the plurality of processors rewrites state information, representing the processing to be performed next, corresponding to the processed data as claimed in claims 3 and 13. Rather, *Kuo et al.* merely discloses that the host computer can store the next set of graphic parameters while a current command is being executed by the processing unit.

F. Rejection of Claims 4 and 14

As discussed above, *Kuo et al.* merely discloses a CPU 14 and a processing unit 22. Nothing in *Kuo et al.* shows, teaches or suggests further comprising a first controller for controlling the plurality of processors to execute a series of processing functions based on state information as claimed in claims 4 and 14. In other words, as claimed in claims 4 and 14, in addition to the plurality of processors and memory, additionally provided is a first controller. Nothing in *Kuo et al.* shows, teaches or suggests this feature.

G. Rejection of Claims 5 and 15

For the reasons as set forth above, nothing in *Kuo et al.* shows, teaches or suggests in addition to the plurality of processors and memory, a first controller as claimed in claims 5 and 15, *Kuo et al.* also does not show, teach or suggest that the first controller rewrites the state information as claimed in claims 5 and 15. Rather, *Kuo et al.* merely discloses the CPU 14 stores the next set of graphic contexts parameters in shared memory 34 while processing unit 22 converts the graphics context parameters into display data. Thus, nothing in *Kuo et al.* shows, teaches or suggests a) a first controller and b) the first controller rewrites state information in response to completion of each processing by the plurality of processors as claimed in claims 5 and 15.

H. Rejection of Claims 6 and 16

As discussed above, *Kuo et al.* merely discloses CPU 14 and processing unit 22. Nothing in *Kuo et al.* shows, teaches or suggests in addition to a plurality of processors and a memory, a second controller for determining an attribute of data to be processed where the second controller rewrites the state information in order to change the order of executing the series of processing functions based on the prescribed attribute as claimed in claims 6 and 16. *Kuo et al.* merely discloses CPU 14 and processing unit 22 and no additional controller is shown, taught or suggested.

Applicants respectfully traverse the Examiner's statement that mask register is a controller. As described in *Kuo et al.*, the mask register is part of the graphics

context 24 which stores parameters needed to carry out a current graphics command. Thus, the mask register is merely a storage unit and thus does not determine an attribute nor can it rewrite state information. Therefore, nothing in *Kuo et al.* shows, teaches or suggests a second controller as claimed in claims 6 and 16.

I. Rejection of Claims 7 and 17

As discussed above, *Kuo et al.* merely discloses a mask register which is part of the graphics context 24 storing information. Nothing in *Kuo et al.* shows, teaches or suggests a second controller which rewrites state information as claimed in claims 7 and 17.

VIII. Conclusion

For all of the above stated reasons, applicants respectfully request the Honorable Board of Patent Appeals and Interferences reverses the Examiner's decision in this application, since applicants respectfully submit that the final rejection of claims 1-26 under 35 U.S.C. §102(b) is in error.

In the event that this paper is not timely filed within the currently set shortened statutory period, Applicants respectfully petition for an appropriate extension of time. The fees for such extension of time may be charged to our Deposit Account No. 02-4800.

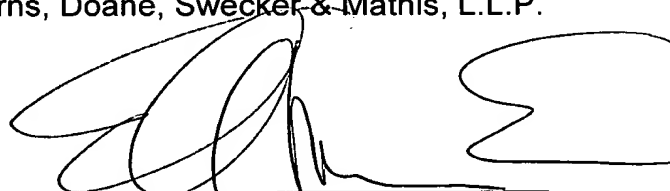
In the event that any additional fees are due with this paper, please charge
our Deposit Account No. 02-4800.

Respectfully submitted,

Burns, Doane, Swecker & Mathis, L.L.P.

Date January 27, 2005

By:

A handwritten signature in black ink, appearing to read 'Ellen Marcie Enas', written over a horizontal line.

Ellen Marcie Enas
Registration No. 32,131

P.O. Box 1404
Alexandria, Virginia 22313-1404
(703) 836-6620



CLAIMS - APPENDIX A

The Appealed Claims

1. (Previously Presented) A data processing system comprising:

a plurality of processors for executing a series of different types of processing functions on data to be processed in a prescribed order, each processor executing a processing function different from one another and said data to be processed being image data that consists of a plurality of pixel data; and

a memory for storing said data to be processed in association with state information to represent the processing to be performed next for each pixel data of said data to be processed, wherein

said processing functions are asynchronously executed on said data to be processed by said plurality of processors, one processing is executed on each pixel data by one of the processors at a time and said plurality of processors share said memory.

2. (Original) The data processing system according to claim 1, wherein

said plurality of processors each determine if said data to be processed can be processed based on said state information.

3. (Previously Presented) The data processing system according to claim 2, wherein

said plurality of processors each execute a processing on said data to be processed, and then rewrite said state information corresponding to the processed data.

4. (Previously Presented) The data processing system according to claim 1, further comprising a first controller for controlling said plurality of processors to execute said series of processing functions based on said state information.

5. (Previously Presented) The data processing system according to claim 4, wherein

said first controller rewrites said state information corresponding to processed data in response to the completion of each processing by said plurality of processors.

6. (Previously Presented) The data processing system according to claim 1, further comprising a second controller for determining an attribute of said data to be processed, wherein

said second controller rewrites said state information corresponding to said data to be processed in order to change the order of executing said series of processing functions if it is determined that said data to be processed has a prescribed attribute.

7. (Previously Presented) The data processing system according to claim 6, wherein

said second controller rewrites said state information corresponding to said data to be processed in order to change the order of executing said series of processing functions if it is determined that said data to be processed has a prescribed attribute.

8. (Previously Presented) The data processing system according to claim 1, wherein

said memory has one region to store said state information corresponding to a single region where said data to be processed is stored.

9. (Previously Presented) The data processing system according to claim 1, wherein

said memory has one region to store said state information corresponding to a plurality of regions where said data to be processed is stored.

10. (Previously Presented) The data processing system according to claim 1, wherein

said data to be processed is image data.

11. (Previously Presented) A data processing system comprising:

a plurality of processing means for executing a series of processing functions of different types on data to be processed in a prescribed order, each processing means executing a processing function different from one another and said data to be processed being image data that consists of a plurality of pixel data; and

memory means for storing said data to be processed in association with state information to represent the processing to be performed next for each pixel data of said data to be processed, wherein

said processing functions are executed asynchronously on said data to be processed by said plurality of processing means, one processing is executed on each pixel data by one of the processing means at a time, and said plurality of processing means share said memory means.

12. (Original) The data processing system according to claim 11, wherein

said plurality of processing means each determine whether said data to be processed can be processed based on said state information.

13. (Previously Presented) The data processing system according to claim 12, wherein

said plurality of processing means each execute a processing on said data to be processed and then rewrite said state information corresponding to the processed data.

14. (Previously Presented) The data processing system according to claim 11, further comprising first control means for controlling said plurality of processing means to execute said series of processing functions based on said state information.

15. (Previously Presented) The data processing system according to claim 14, wherein

said first control means rewrites said state information corresponding to processed data in response to the completion of each processing by said plurality of processing means.

16. (Previously Presented) The data processing system according to claim 11, further comprising a second control means for determining an attribute of said data to be processed, wherein

if it is determined that said data to be processed has a prescribed attribute, said second control means rewrites said state information corresponding to said data to be processed in order to change the order of executing said series of processing functions.

17. (Previously Presented) The data processing system according to claim 16, wherein

said second control means rewrites said state information corresponding to said data to be processed in order to remove a part of said series of processing functions if it is determined that said data to be processed has a prescribed attribute.

18. (Previously Presented) The data processing system according to claim 11, wherein

said memory means has one region to store said state information corresponding to a single region where said data to be processed is stored.

19. (Previously Presented) The data processing system according to claim 11, wherein

said memory means has one region to store said state information corresponding to a plurality of regions where said data to be processed is stored.

20. (Original) The data processing system according to claim 11, wherein said data to be processed is image data.

21. (Previously Presented) The data processing system of claim 1 wherein a given data item is stored at the same location in said memory after each of said plurality of processing functions is performed on said given data item.

22. (Previously Presented) The data processing system of claim 21 wherein the state information for said given data item is stored at the same location in said memory after each of said plurality of processing functions is performed on said given data item.

23. (Previously Presented) The data processing system of claim 11 wherein a given data item is stored at the same location in said memory means after each of said plurality of processing functions is performed on said given data item.

24. (Previously Presented) The data processing system of claim 23 wherein the state information for said given data item is stored at the same location in said memory means after each of said plurality of processing functions is performed on said given data item.

25. (Previously Presented) A data processing device comprising:
a first processor for executing first processing on data to be processed;
a second processor for executing second processing on said data to be processed that was subjected to the first processing; and
a memory for storing said data to be processed in association with state information to represent the processing state of said data, wherein
said first and second processing are asynchronously executed on said data to be processed by said first and second processors and said first and second processors share said memory.

26. (Previously Presented) An image processing device comprising:

- a first image processor for executing first image processing on image data;
- a second image processor for executing second image processing on said image data that was subjected to the first image processing; and
- a memory for storing said image data in association with state information to represent the processing state of said image data, wherein

said first and second image processings are asynchronously executed on said image data by said first and second image processors and said first and second image processors share said memory.

FIGURES 1-19 – APPENDIX B



FIG. 1

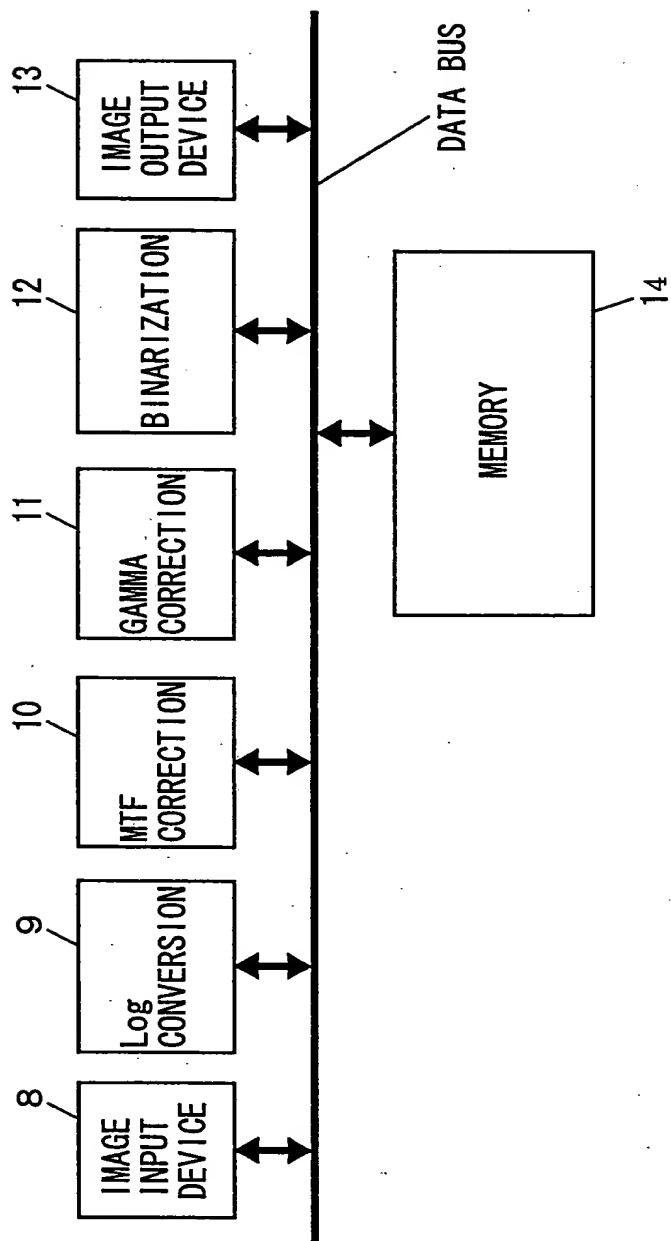


FIG. 2

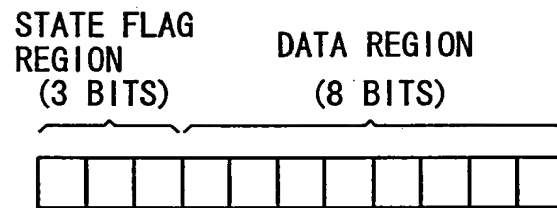


FIG. 3

PROCESSING	PROCESSIBLE FLAG	PROCESSED FLAG
Log CONVERSION	0 0 0	0 0 1
MTF CORRECTION	0 0 1	0 1 0
GAMMA CORRECTION	0 1 0	0 1 1
BINARIZATION	0 1 1	1 1 0
OUTPUT	1 1 0	1 1 1

FIG. 4

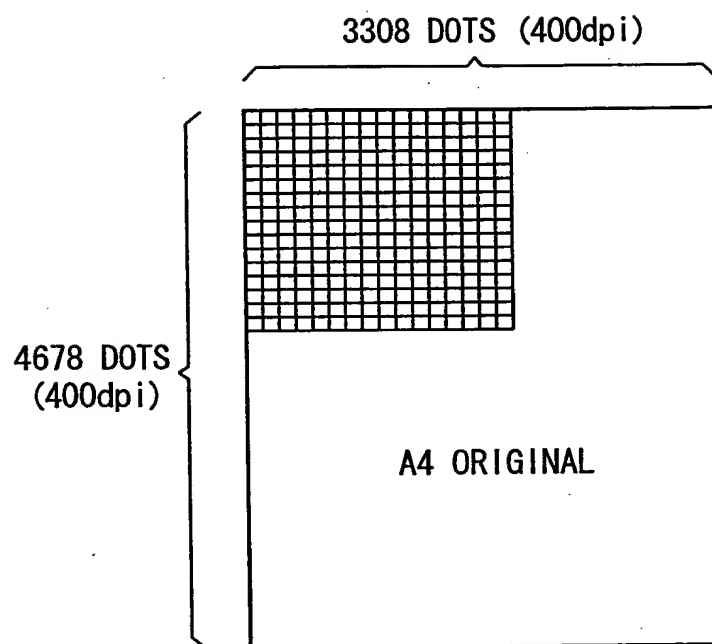


FIG. 5A FIG. 5B FIG. 5C FIG. 5D FIG. 5E

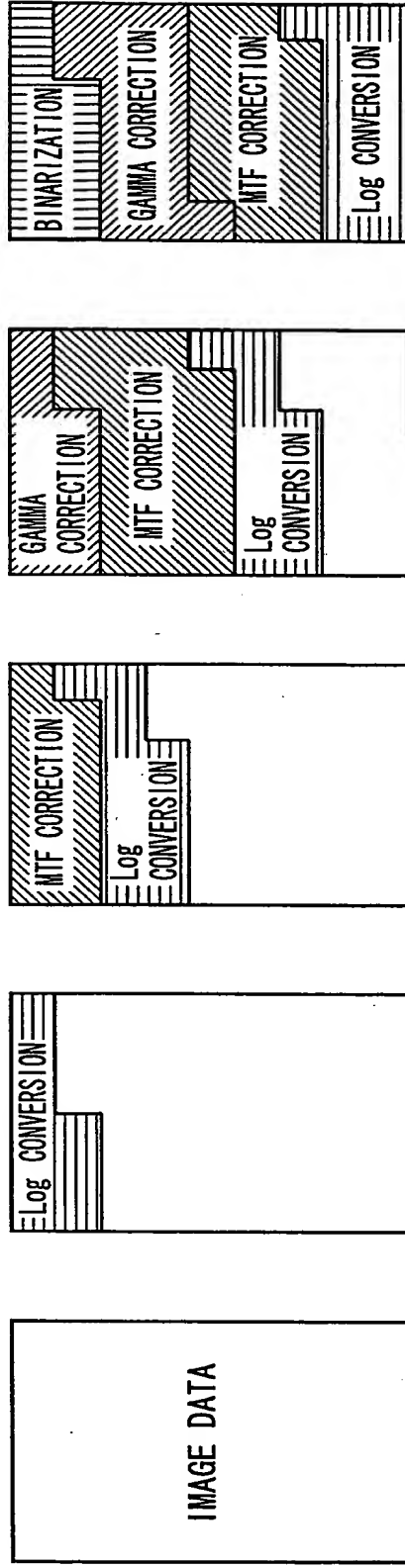


FIG. 5F

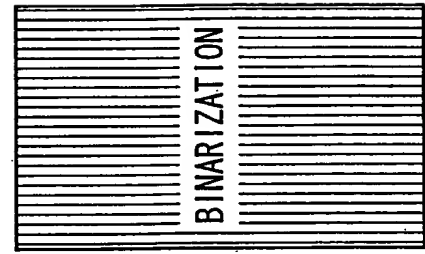


FIG. 6

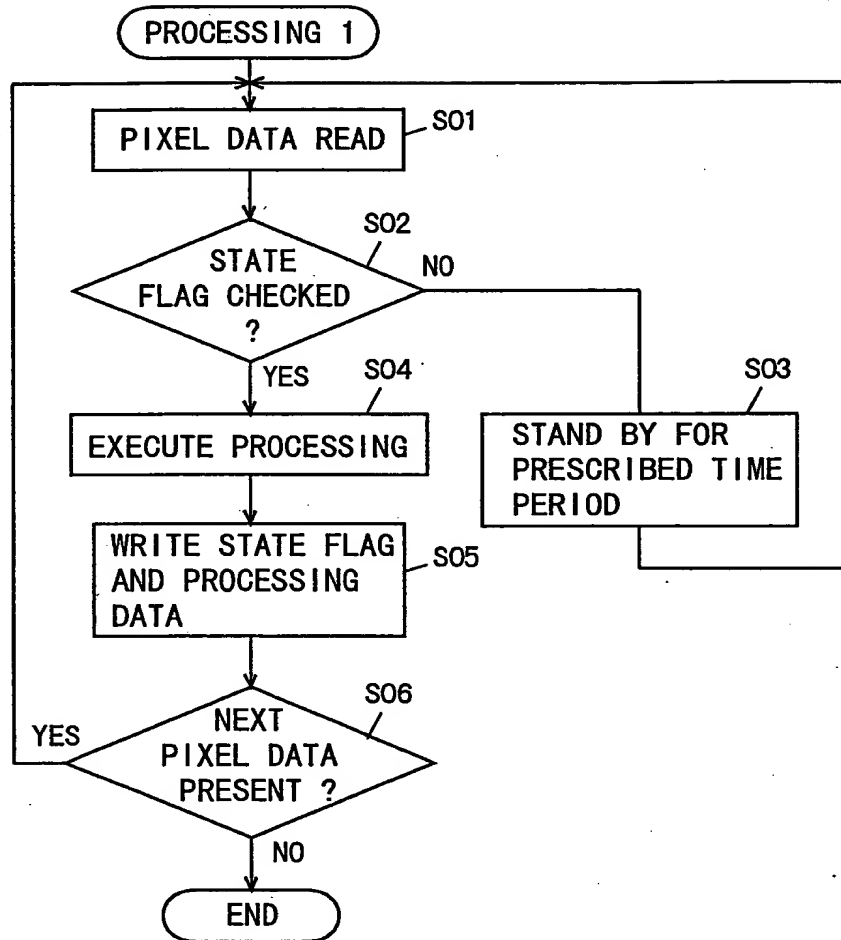


FIG. 7A

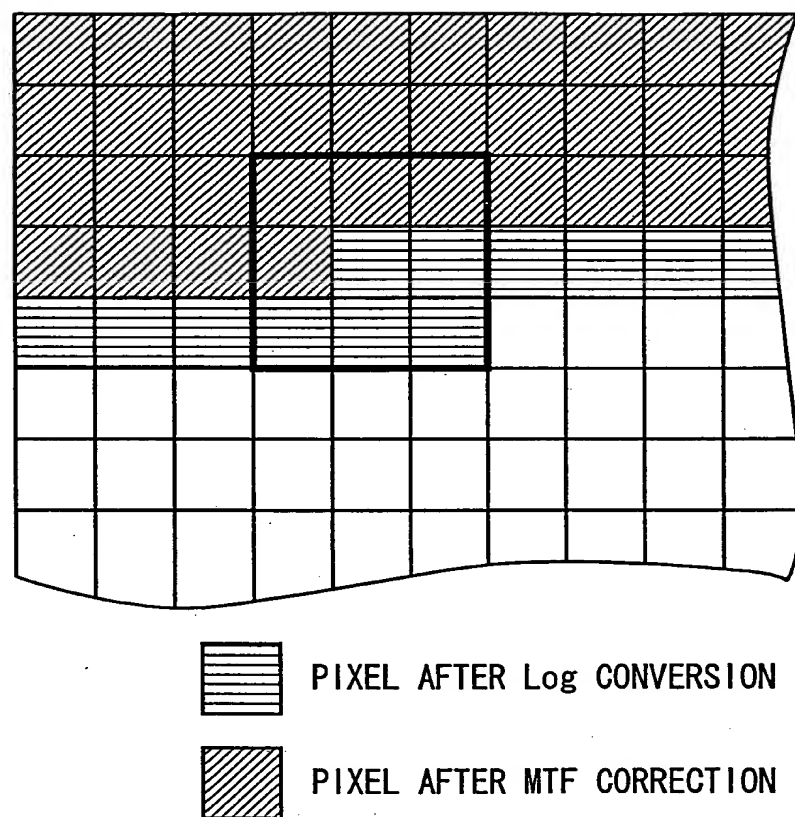


FIG. 7B

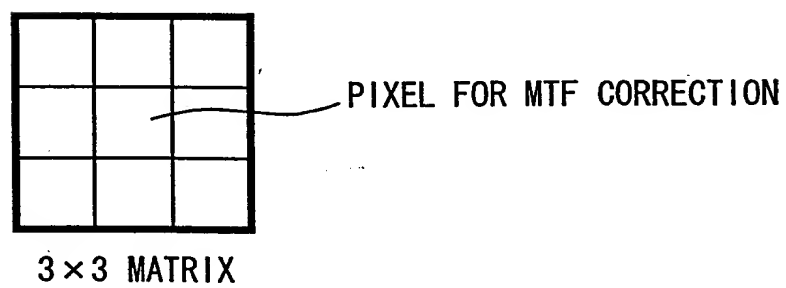


FIG. 8

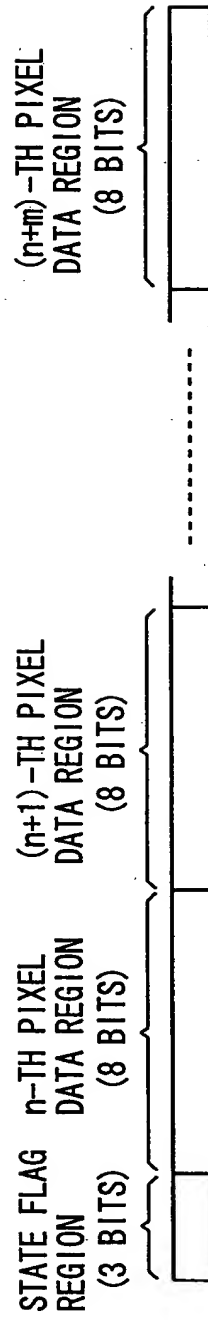


FIG. 9

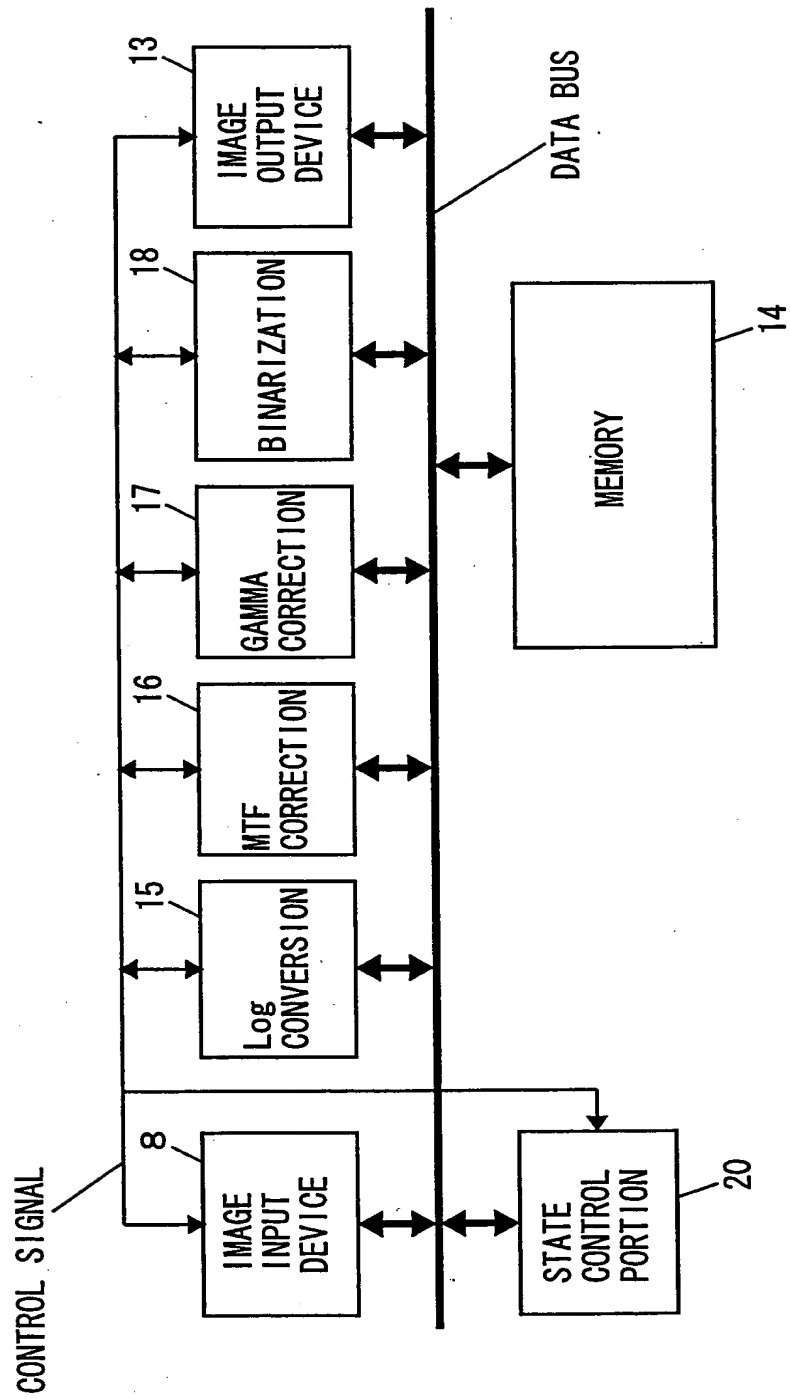


FIG. 10

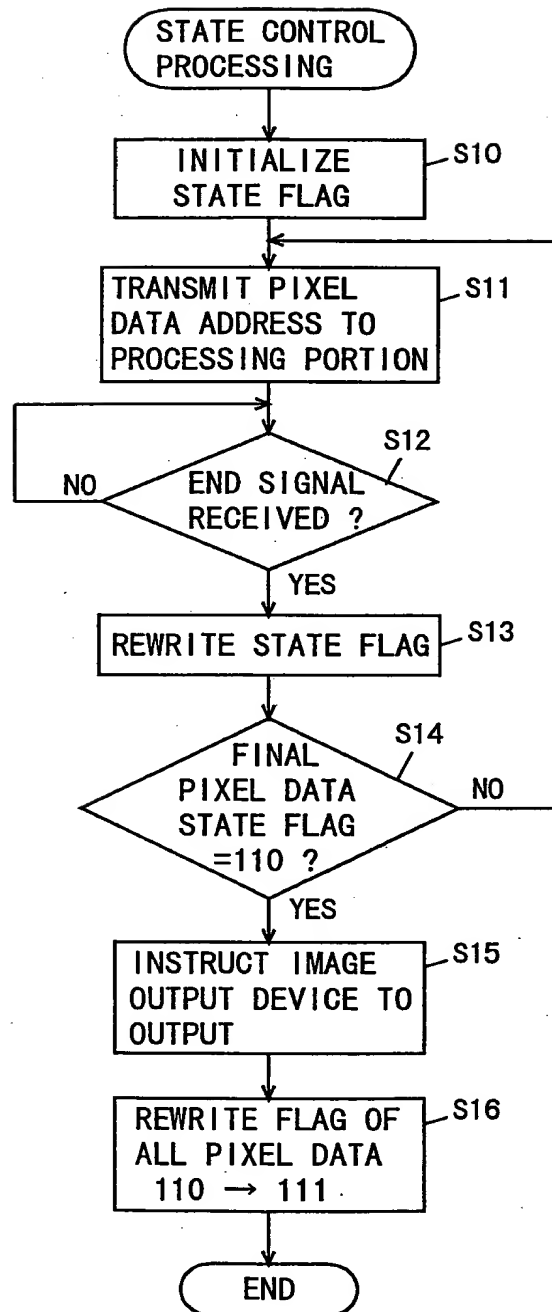


FIG. 11

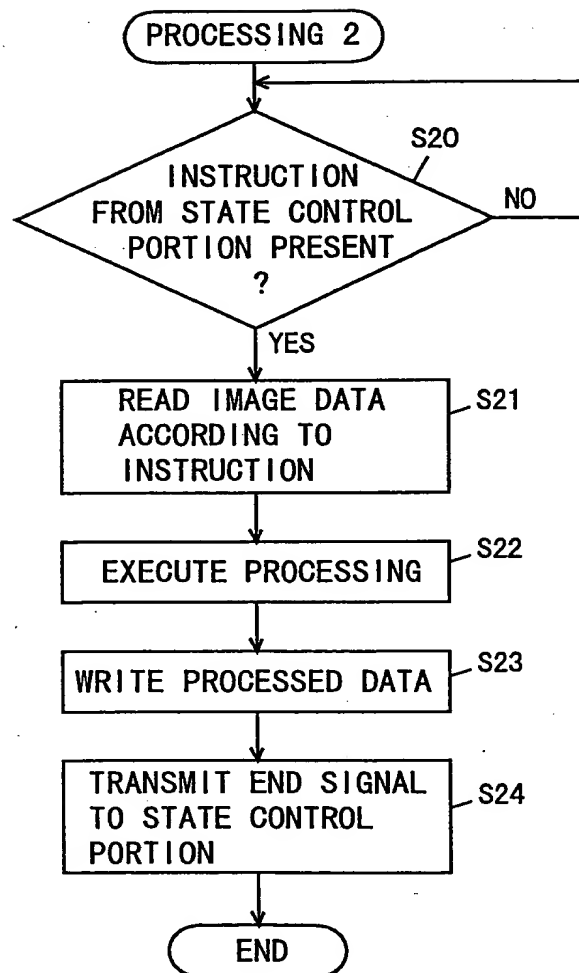


FIG. 12

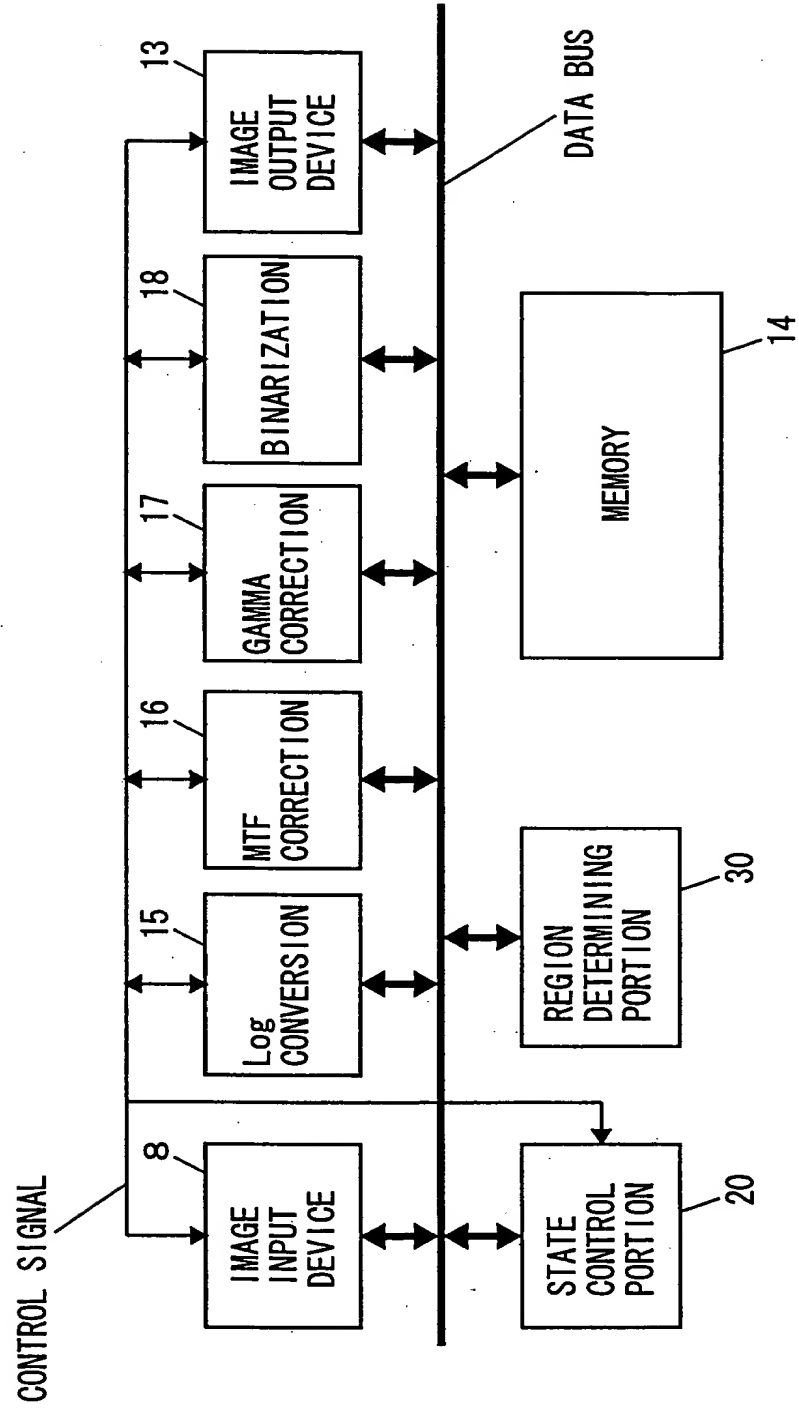
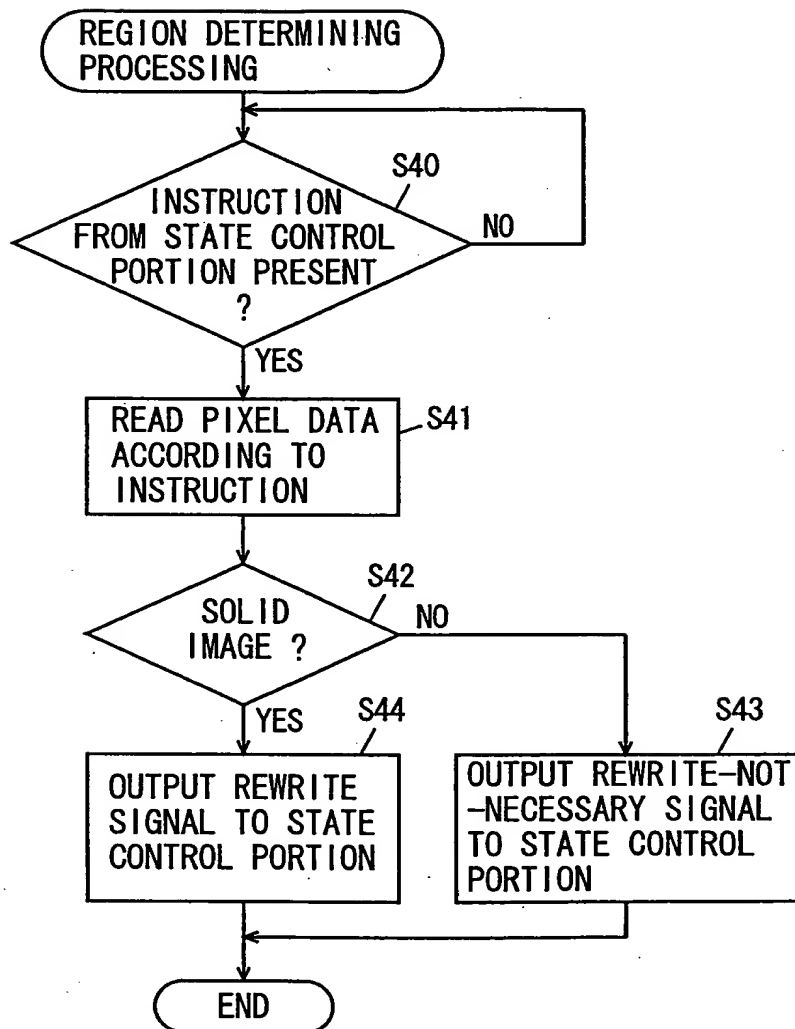


FIG. 13



F I G. 1 4

PROCESSING	PROCESSIBLE FLAG	PROCESSED FLAG
REGION DETERMINING	0 0 0	0 0 1
		1 0 0
Log CONVERSION	0 0 1	0 1 0
MTF CORRECTION	0 1 0	0 1 1
GAMMA CORRECTION	0 1 1	1 0 0
BINARIZATION	1 0 0	1 1 0
OUTPUT	1 1 0	1 1 1

FIG. 15

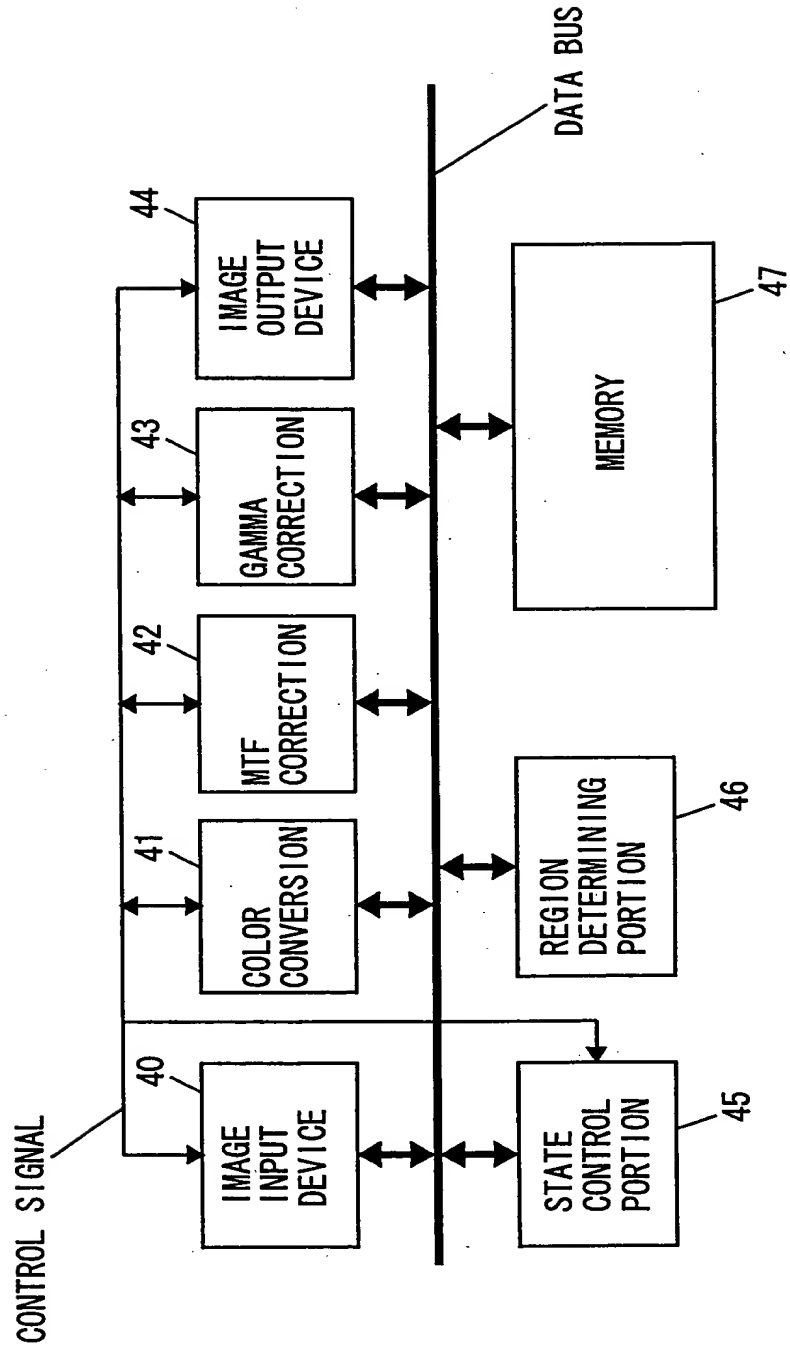


FIG. 16

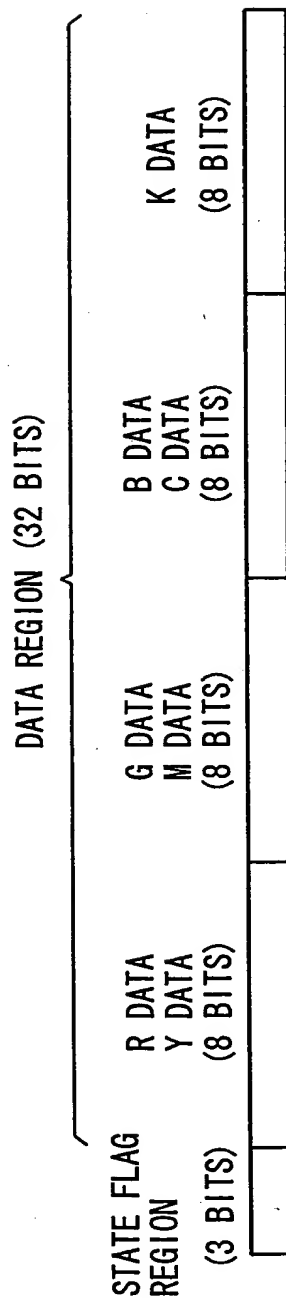


FIG. 17

PROCESSING	PROCESSIBLE FLAG	PROCESSED FLAG
REGION DETERMINING	0 0 0	0 0 1
		1 0 1
COLOR CONVERSION	0 0 1	0 1 0
	1 0 1	1 1 0
MTF CORRECTION	0 1 0	0 1 1
GAMMA CORRECTION	0 1 1	1 1 0
OUTPUT	1 1 0	1 1 1

FIG. 18

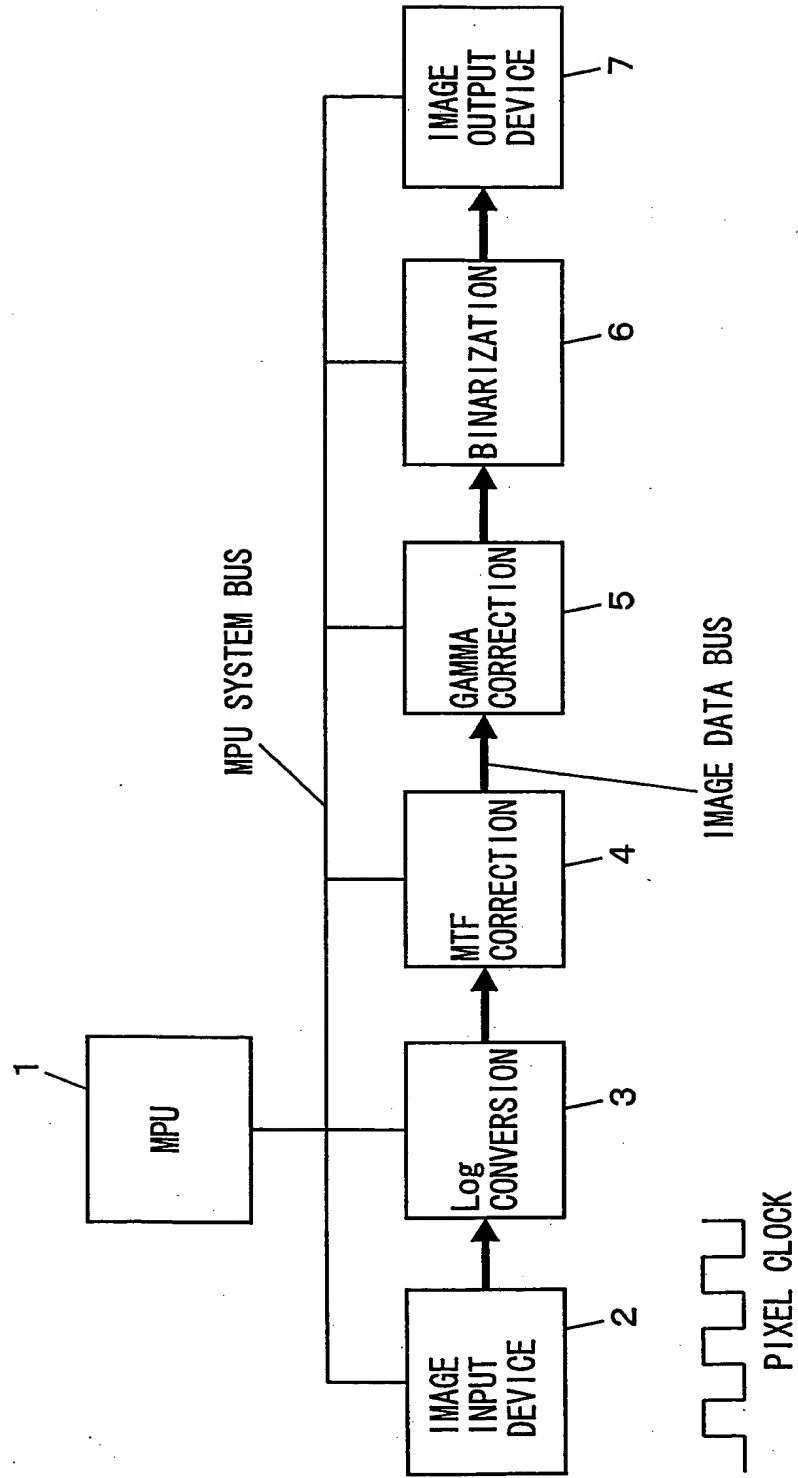


FIG. 19

